



- (51) **International Patent Classification:**
G06F 3/06 (2006.01) *G06F 12/00* (2006.01)
- (21) **International Application Number:**
PCT/IB2012/052564
- (22) **International Filing Date:**
22 May 2012 (22.05.2012)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
13/113,949 23 May 2011 (23.05.2011) US
- (71) **Applicant (for all designated States except US):** **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).
- (71) **Applicants (for MG only):** **IBM UNITED KINGDOM LIMITED** [GB/GB]; PO Box 41, North Harbour, Portsmouth Hampshire PO6 3AU (GB). **IBM JAPAN LIMITED** [JP/JP]; 19-21 Nihonbashi, Hakozaiki-cho, Chuo-ku, Tokyo 103-8510 (JP).
- (72) **Inventors; and**
- (75) **Inventors/Applicants (for US only):** **BENHASE, Michael, Thomas** [US/US]; IBM Corporation, M/P 9032-1 102, 9000 South Rita Road, Tucson, Arizona 85744-0002 (US). **GUPTA, Lokesh, Mohan** [US/US]; IBM Corporation, M/P 9032-1 103, 9000 South Rita Road, Tucson, Arizona 85744-0002 (US). **KALOS, Matthew, Joseph**

[US/US]; IBM Corporation, M/P 9032-1 102, 9000 South Rita Road, Tucson, Arizona 85744-0002 (US). **GILL, Binny, Sher** [US/US]; IBM Corporation, Almaden Research Center, Intellectual Property Law C4TA/J2B, 650 Harry Road, San Jose, California 95120-6099 (US).

(74) **Agent:** **STRETTON, Peter;** IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester Hampshire SO21 2JN (GB).

(81) **Designated States (unless otherwise indicated, for every kind of national protection available):** AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States (unless otherwise indicated, for every kind of regional protection available):** ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) **Title:** CACHING DATA IN A STORAGE SYSTEM HAVING MULTIPLE CACHES

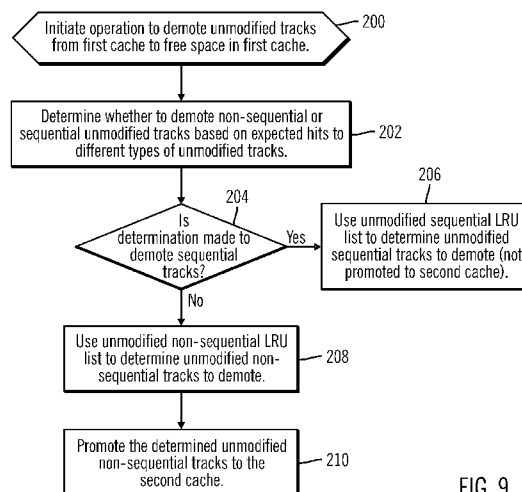


FIG. 9

(57) **Abstract:** Provided are a computer program product, system, and method for using an attribute of a write request to determine where to cache data in a storage system having multiple caches including non-volatile storage cache in a sequential access storage device. Received modified tracks are cached in the non-volatile storage device integrated with the sequential access storage device in response to determining to cache the modified tracks. A write request having modified tracks is received. A determination is made as to whether an attribute of the received write request satisfies a condition. The received modified tracks for the write request are cached in the non-volatile storage device in response to determining that the determined attribute does not satisfy the condition. A destage request is added to a request queue for the received write request having the determined attribute not satisfying the condition.

WO 2012/160514 A1

Published:

— with international search report (Art. 21(3))

— before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))

CACHING DATA IN A STORAGE SYSTEM HAVING MULTIPLE CACHES

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to a computer program product, system, and method for using an attribute of a write request to determine where to cache data in a storage system having multiple caches including non-volatile storage cache in a sequential access storage device.

Description of the Related Art

A cache management system buffers tracks in a storage device recently accessed as a result of read and write operations in a faster access storage device, such as memory, than the storage device storing the requested tracks. Subsequent read requests to tracks in the faster access cache memory are returned at a faster rate than returning the requested tracks from the slower access storage, thus reducing read latency. The cache management system may also return complete to a write request when the modified track directed to the storage device is written to the cache memory and before the modified track is written out to the storage device, such as a hard disk drive. The write latency to the storage device is typically significantly longer than the latency to write to a cache memory. Thus, using cache also reduces write latency.

A cache management system may maintain a linked list having one entry for each track stored in the cache, which may comprise write data buffered in cache before writing to the storage device or read data. In the commonly used Least Recently Used (LRU) cache technique, if a track in the cache is accessed, i.e., a cache "hit", then the entry in the LRU list for the accessed track is moved to a Most Recently Used (MRU) end of the list. If the requested track is not in the cache, i.e., a cache miss, then the track in the cache whose entry is at the LRU end of the list may be removed (or destaged back to storage) and an entry for the track data staged into cache from the storage is added to the MRU end of the LRU list. With this LRU cache technique, tracks that are more frequently accessed are likely to remain

in cache, while data less frequently accessed will more likely be removed from the LRU end of the list to make room in cache for newly accessed tracks.

The LRU cache technique seeks to optimize for temporal locality so as to destage tracks that are least likely to be rewritten soon in order to minimize the number of destage operations, i.e., if a write that is not destaged is overwritten then the destaging of the overwritten write is avoided, thus saving the time and effort of writing the data from cache to disk. On the other hand there is also a desire to destage in a manner that exploits spatial locality, which means that data is written to storage locations that are closest to each other to minimize the distance the storage device write mechanism and storage media needs to be moved to reach the next storage location to write.

One technique for exploiting both temporal and spatial locality is the Wise Ordering for Writes (WOW) algorithm. The WOW algorithm employs a circular linked list or clock where the circular linked list has one entry for each write request buffered in cache. The entries are ordered in the linked list according to the storage location to which the associated write request is directed to exploit the benefits of spatial locality. Further, each entry includes a bit indicating whether the write data for the storage location in the cache has been recently updated. The bit for an entry is set when the write data for the entry is updated. A pointer points to a current entry in the circular linked list. A task using the WOW algorithm accesses an entry addressed by the pointer. If the bit for the entry indicates that the data for the entry in cache has been recently updated, then the bit is set to indicate that the write data has not been recently updated and the pointer incremented to point to the next entry so that the entry having write data to a storage location next closest in spatial proximity to the previously written storage location is considered. The entry is selected to write that is closest in spatial proximity to the last written storage location and whose bit indicates that the write data for the entry has not recently been updated.

Thus, with the WOW algorithm, spatial locality is exploited because a next entry to write is selected for consideration that is closest in spatial proximity to the last destaged write request. Further, temporal locality is exploited because an entry that has recently been written will be skipped until the pointer circles back to that skipped entry to consider.

Disk drives may implement the WOW algorithm and other algorithms that take both the linear and the angular position of the write tracks into account and optimize for both with respect to a current write head position to determine the minimal total service time. This process is referred to as “command re-ordering based on seek and rotational optimization”. The disk drive logic boards will analyze write requests and determine which to do first based on both how much time will be required to seek to the various cylinders and angular position of the track to write, and how much time will elapse waiting for the data to rotate under the heads.

There is a need in the art for improved techniques for using cache in a storage system.

SUMMARY

Provided are a computer program product, system, and method for using an attribute of a write request to determine where to cache data in a storage system having multiple caches including non-volatile storage cache in a sequential access storage device. Received modified tracks are cached in the non-volatile storage device integrated with the sequential access storage device in response to determining to cache the modified tracks. A write request having modified tracks is received. A determination is made as to whether an attribute of the received write request satisfies a condition. The received modified tracks for the write request are cached in the non-volatile storage device in response to determining that the determined attribute does not satisfy the condition. A destage request is added to a request queue for the received write request having the determined attribute not satisfying the condition. The received modified tracks for the write request having the determined attribute satisfying the condition are written at a higher priority than modified tracks for write requests having the attribute not satisfying the condition.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates an embodiment of a computing environment.

FIG. 2 illustrates an embodiment of first cache management information.

FIG. 3 illustrates an embodiment of second cache management information.

FIG. 4 illustrates an embodiment of a sequential access storage device.

FIG. 5 illustrates an embodiment of a first cache control block.

FIG. 6 illustrates an embodiment of a second cache control block.

FIG. 7 illustrates an embodiment of a non-volatile storage cache control block.

FIG. 8 illustrates an embodiment of a spatial index entry.

FIG. 9 illustrates an embodiment of operations to determine whether to remove tracks in the first cache to free space for tracks to add to the first cache.

FIG. 10 illustrates an embodiment of operations to free space in the first cache.

FIG. 11 illustrates an embodiment of operations to add a track to the first cache.

FIG. 12 illustrates an embodiment of operations to promote a track to the second cache.

FIG. 13 illustrates an embodiment of operations to free space in the second cache.

FIG. 14 illustrates an embodiment of operations to process a read request for requested tracks.

FIG. 15 illustrates an embodiment of operations at the sequential access storage device to process a write request.

FIG. 16 illustrates an embodiment of operations at the sequential access storage device to determine whether to cache the modified tracks for a write request in a non-volatile storage device.

FIG. 17 illustrates an embodiment of operations at the sequential access storage device to process a request queue.

FIG. 18 illustrates an embodiment of operations at the sequential access storage device to process a destage request in the request queue.

FIG. 19 illustrates an embodiment of operations at the sequential access storage device to process write requests in the request queue and the write bypass queue.

DETAILED DESCRIPTION

FIG. 1 illustrates an embodiment of a computing environment. A plurality of hosts 2a, 2b...2n may submit Input/Output (I/O) requests to a storage controller 4 over a network 6 to

access data at volumes 8 (e.g., Logical Unit Numbers, Logical Devices, Logical Subsystems, etc.) in a storage 10. The storage controller 4 includes a processor complex 12, including one or more processors with single or multiple cores, a first cache 14, a first cache backup device 16, to backup tracks in the cache 14, and a second cache 18. The first 14 and second 18 caches cache data transferred between the hosts 2a, 2b...2n and the storage 10. The first cache backup device 16 may provide non-volatile storage of tracks in the first cache 14. In a further embodiment, the first cache backup device 16 may be located in a cluster or hardware on a different power boundary than that of the first cache 14.

The storage controller 4 has a memory 20 that includes a storage manager 22 for managing the transfer of tracks transferred between the hosts 2a, 2b...2n and the storage 10 and a cache manager 24 that manages data transferred between the hosts 2a, 2b...2n and the storage 10 in the first cache 14, first cache backup device 16, and the second cache 18. A track may comprise any unit of data configured in the storage 10, such as a track, Logical Block Address (LBA), etc., which is part of a larger grouping of tracks, such as a volume, logical device, etc. The cache manager 24 maintains first cache management information 26 and second cache management information 28 to manage read (unmodified) and write (modified) tracks in the first cache 14 and the second cache 18. A first cache backup device index 30 provides an index of track identifiers to a location in the first cache backup device 16.

The storage manager 22 and cache manager 24 are shown in FIG. 1 as program code loaded into the memory 20 and executed by the processor complex 12. Alternatively, some or all of the functions may be implemented in hardware devices in the storage controller 4, such as in Application Specific Integrated Circuits (ASICs).

The second cache 18 may store tracks in a log structured array (LSA) 32, where tracks are written in a sequential order as received, thus providing a temporal ordering of the tracks written to the second cache 18. In a LSA, later versions of tracks already present in the LSA are written at the end of the LSA 32. In alternative embodiments, the second cache 18 may store data in formats other than in an LSA.

In one embodiment, the first cache 14 may comprise a Random Access Memory (RAM), such as a Dynamic Random Access Memory (DRAM), and the second cache 18 may comprise a flash memory, such as a solid state device, and the storage 10 is comprised of one or more sequential access storage devices, such as hard disk drives and magnetic tape. The storage 10 may comprise a single sequential access storage device or may comprise an array of storage devices, such as a Just a Bunch of Disks (JBOD), Direct Access Storage Device (DASD), Redundant Array of Independent Disks (RAID) array, virtualization device, etc. In one embodiment, the first cache 14 is a faster access device than the second cache 18, and the second cache 18 is a faster access device than the storage 10. Further, the first cache 14 may have a greater cost per unit of storage than the second cache 18 and the second cache 18 may have a greater cost per unit of storage than storage devices in the storage 10.

The first cache 14 may be part of the memory 20 or implemented in a separate memory device, such as a DRAM. In one embodiment, the first cache backup device 16 may comprise a non-volatile backup storage (NVS), such as a non-volatile memory, e.g., battery backed-up Random Access Memory (RAM), static RAM (SRAM), etc.

The network 6 may comprise a Storage Area Network (SAN), a Local Area Network (LAN), a Wide Area Network (WAN), the Internet, and Intranet, etc.

FIG. 2 illustrates an embodiment of the first cache management information 26 including a track index 50 providing an index of tracks in the first cache 14 to control blocks in a control block directory 52; an unmodified sequential LRU list 54 providing a temporal ordering of unmodified sequential tracks in the first cache 14; a modified LRU list 56 providing a temporal ordering of modified sequential and non-sequential tracks in the first cache 14; and an unmodified non-sequential LRU list 58 providing a temporal ordering of unmodified non-sequential tracks in the first cache 14.

In certain embodiments, upon determining that the first cache backup device 16 is full, the modified LRU list 56 is used to destage modified tracks from the first cache 14 so that the copy of those tracks in the first cache backup device 16 may be discarded to make room in the first cache backup device 16 for new modified tracks.

FIG. 3 illustrates an embodiment of the second cache management information 28 including a track index 70 providing an index of tracks in the second cache 18 to control blocks in a control block directory 72; an unmodified list 74 providing a temporal ordering of unmodified tracks in the second cache 18; and a spatial index 76 providing a spatial ordering of the modified tracks in the second cache 18 based on the physical locations in the storage 10 at which the modified tracks are stored.

All the LRU lists 54, 56, 58, and 74 may include the track IDs of tracks in the first cache 14 and the second cache 18 ordered according to when the identified track was last accessed. The LRU lists 54, 56, 58, and 74 have a most recently used (MRU) end indicating a most recently accessed track and a LRU end indicating a least recently used or accessed track. The track IDs of tracks added to the caches 14 and 18 are added to the MRU end of the LRU list and tracks demoted from the caches 14 and 18 are accessed from the LRU end. The track indexes 50 and 70 and spatial index 76 may comprise a scatter index table (SIT). Alternative type data structures may be used to provide the temporal ordering of tracks in the caches 14 and 18 and spatial ordering of tracks in the second cache 18.

Non-sequential tracks may comprise Online Line Transaction Processing (OLTP) tracks, which often comprise small block writes that are not fully random and have some locality of reference, i.e., have a probability of being repeatedly accessed.

FIG. 4 illustrates an embodiment of a sequential access storage device 100, where the storage 10 may be implemented with one or multiple sequential access storage devices 100. The sequential access storage device 100 includes control logic shown as the I/O manager 102, a non-volatile storage device 104 to buffer modified data, and a memory 106 including a track index 108 providing an index of tracks in the non-volatile storage device 104 to control blocks in a control block directory 110; a spatial index 112 providing a spatial ordering of the modified tracks in the non-volatile storage 104 on the physical locations in a sequential access storage medium 114 at which the modified tracks are stored; and a request queue 116 in which read and write requests are queued. The I/O manager 102 adds read and write request to the request queue 112, and accesses read and write requests from the request queue 112 to execute against a sequential access medium 114. The I/O manager 102 may

send commands to a read/write control unit 118 that generates control signals to move one or more actuators having read/write heads 120 to a position on the sequential access storage medium 114 at which data can be read or written.

The memory 106 further includes a write bypass queue 118 to buffer sequential write requests and their modified tracks in a buffer 122 that will not be cached in the non-volatile storage device 104, but are directly written to the sequential access storage medium 114. The buffer 122 may temporarily buffer read and write input requests and data being returned to a read request. The buffer 122 may be in a separate device than the non-volatile storage device 104 and may comprise smaller storage space than available in the non-volatile storage device 104.

A buffer 124 in the device 100 may temporarily buffer read and write input requests and data being returned to a read request. The buffer 124 may also be used to temporarily buffer modified tracks for write requests not maintained in the non-volatile storage device, such as for sequential write requests and their modified data. The buffer 124 may be in a separate device than the non-volatile storage device 104 and may comprise smaller storage space than available in the non-volatile storage device 104. Alternatively, some or all of the buffer 124 may be implemented in the non-volatile storage device.

The sequential access storage medium 114 may comprise one or more hard disk drive platters for a hard disk drive device or magnetic tape. In certain embodiments, the non-volatile storage device 104 may comprise a flash memory device comprised of solid state storage. In certain embodiments, the non-volatile storage device 104, e.g., flash memory, is implemented on the sequential access storage device 100 circuit board within the enclosure including the sequential access storage device 100 components. For instance, the may comprise an 8GB flash memory device.

Some or all of the functions of the I/O manager 102 may be implemented as code executed by a processor in the sequential access storage device 100. Alternatively, some or all of the functions of the I/O manager 102 may be implemented in an ASIC on the sequential access storage device 100.

FIG. 5 illustrates an embodiment of a first cache control block 150 entry in the control block directory 52, including a control block identifier (ID) 152, a first cache location 154 of the physical location of the track in the first cache 14, information 156 indicating whether the track is modified or unmodified, and information 158 indicating whether the track is a sequential or non-sequential access.

FIG. 6 illustrates an embodiment of a second cache control block 160 entry in the second cache control block directory 72, including a control block identifier (ID) 162 and an LSA location 164 where the track is located in the LSA 32.

FIG. 7 illustrates an embodiment of a non-volatile storage control block 170 entry in the non-volatile storage 104 control block directory 110, including a control block identifier (ID) 172 and a physical location 174 at which the track is located, such as an LSA location if the track is stored in a LSA on the non-volatile storage device.

FIG. 8 illustrates a spatial index entry 180 including a track identifier 182 of a track in the non-volatile storage device 104 and the physical location 184 of where the track is stored in the sequential access storage medium 114, such as a cylinder, platter number, angular position on the cylinder, etc.

FIG. 9 illustrates an embodiment of operations performed by the cache manager 24 to demote unmodified tracks from the first cache 14. The demote operation may be initiated upon determining to free space in the first cache 14. Upon initiating (at block 200) an operation to determine whether to remove tracks from the first cache 14 to accommodate tracks being added to the first cache 14, the cache manager 24 determines (at block 202) whether to demote non-sequential or sequential unmodified tracks based on expected hits to different types of unmodified tracks. If (at block 204) the determination is to demote unmodified sequential tracks, then the cache manager 24 uses (at block 206) the unmodified sequential LRU list 54 to determine unmodified sequential tracks to demote, from the LRU end of the list, which are not promoted to the second cache 18. If (at block 204) the determination is made to demote unmodified non-sequential tracks, then the cache manager 24 uses the unmodified non-sequential LRU list 58 to determine (at block 208) unmodified

non-sequential tracks to demote. The unmodified non-sequential tracks are promoted (at block 210) to the second cache 18.

FIG. 10 illustrates an embodiment of operations performed by the cache manager 24 to destage modified tracks from the first cache 14. The cache manager 24 may regularly destage tracks as part of scheduled operations and increase the rate of destages if space is needed in the first cache backup device 16. Upon initiating (at block 250) the operation to destage modified tracks, the cache manager 24 processes (at block 252) the modified LRU list 56 to determine modified tracks to destage, from the LRU end of the LRU list 56. The cache manager 24 writes (at block 254) the determined modified tracks (sequential or non-sequential) to the storage 10, bypassing the second cache 18. The cache manager 24 discards (at block 260) the copy of the destaged modified tracks from the first cache backup device 16.

With the operations of FIGs. 9 and 10, non-sequential tracks are demoted but not promoted to the second cache 18. Modified tracks (writes) are written directly to the storage 10, bypassing the second cache. Sequential unmodified tracks (reads) are discarded and not copied elsewhere, and unmodified non-sequential tracks demoted from the first cache 14 are promoted to the second cache 18.

FIG. 11 illustrates an embodiment of operations performed by the cache manager 24 to add, i.e., promote, a track to the first cache 14, which track may comprise a write or modified track from a host 2a, 2b...2n, a non-sequential track in the second cache 18 that is subject to a read request and as a result moved to the first cache 14, or read requested data not found in either cache 14 or 18 and retrieved from the storage 10. Upon receiving (at block 300) the track to add to the first cache 14, the cache manager 24 creates (at block 301) a control block 150 (FIG. 5) for the track to add indicating the 154 location in the first cache 14 and whether the track is modified/unmodified 156 and sequential/non-sequential 158. This control block 150 is added to the control block directory 52 of the first cache 14. The cache manager 24 adds (at block 302) an entry to first cache track index 50 having the track ID of track to add and an index to the created cache control block 150 in the control block directory 52. An entry is added (at block 304) to the MRU end of the LRU list 54, 56 or 58 of the track type

of the track to add. If (at block 306) the track to add is a modified non-sequential track, then the track to add is also copied (at block 308) to the first cache backup device 16 and an entry is added to the first cache backup device index 30 for the added track. If (at block 306) the track to add is unmodified sequential, control ends.

FIG. 12 illustrates an embodiment of operations performed by the cache manager 24 to promote an unmodified non-sequential track to the second cache 18 that is being demoted from the first cache 14. Upon initiating (at block 350) the operation to promote a track to the second cache 18, the cache manager 24 adds (at block 352) the track being promoted to the LSA 32 in the second cache 18 and creates (at block 354) a control block 160 (FIG. 6) for the track to add indicating the track location 164 in the LSA 32. An entry is added (at block 356) to the second cache track index 70 having the track ID of the promoted track and an index to the created cache control block 160 in the control block directory 72 for the second cache 18. The cache manager 24 indicates (at block 360) the promoted track at the MRU end of the unmodified LRU list 74, such as by adding the track ID to the MRU end.

The cache manager 12 may use the second cache 18 as a read-only cache for only unmodified sequential tracks. Modified sequential and non-sequential tracks are written directly to the sequential access storage device 100 and the non-volatile storage device 104 in the sequential access storage device 100 provides a write cache for modified non-sequential tracks.

FIG. 13 illustrates an embodiment of operations performed by the cache manager 24 to free space in the second cache 18 for new tracks to add to the second cache 18, i.e., tracks being demoted from the first cache 14. Upon initiating this operation (at block 400) the cache manager 24 determines (at block 402) unmodified tracks in the second cache 18 from the LRU end of the unmodified LRU list 74 and invalidates (at block 404) the determined unmodified tracks without destaging the invalidated unmodified tracks to the storage 10.

FIG. 14 illustrates an embodiment of operations performed by the cache manager 24 to retrieve requested tracks for a read request from the caches 14 and 18 and storage 10. The storage manager 22 processing the read request may submit requests to the cache manager

24 for the requested tracks. Upon receiving (at block 450) the request for the tracks, the cache manager 24 uses (at block 454) the first cache track index 50 to determine whether all of the requested tracks are in the first cache 14. If (at block 454) all requested tracks are not in the first cache 14, then the cache manager 24 uses (at block 456) the second cache track index 70 to determine any of the requested tracks in the second cache 18 not in the first cache 14. If (at block 458) there are any requested tracks not found in the first 14 and second 18 caches, then the cache manager 24 determines (at block 460) any of the requested tracks in the storage 10, from the second cache track index 70, not in the first 14 and the second 18 caches. The cache manager 24 then promotes (at block 462) any of the determined tracks in the second cache 18 and the storage 10 to the first cache 14. The cache manager 24 uses (at block 464) the first cache track index 50 to retrieve the requested tracks from the first cache 14 to return to the read request. The entries for the retrieved tracks are moved (at block 466) to the MRU end of the LRU list 54, 56, 58 including entries for the retrieved tracks. With the operations of FIG. 13, the cache manager 24 retrieves requested tracks from a highest level cache 14, then second cache 18 first before going to the storage 10, because the caches 14 and 18 would have the most recent modified version of a requested track. The most recent version is first found in the first cache 14, then the second cache 18 if not in the first cache 14 and then the storage 10 if not in either cache 14, 18.

With the operations of FIG. 14, the cache manager 24 gathers requested tracks from a highest level cache 14, then second cache 18 first before going to the storage 10, because the caches 14 and 18 would provide the fastest access to requested tracks and the first cache 14 provides the most recent modified version of a requested track.

FIG. 15 illustrates an embodiment of operations performed by the I/O manager 102 at the sequential access storage device 100 to process a write request with modified tracks for the sequential access storage medium 114. Upon receiving (at block 500) the write request, the I/O manager 102 determines (at block 502) whether an attribute of the write request satisfies a condition. If (at block 502) the condition is satisfied, such as the write request is a sequential write request or the block size of the write request is above a predetermined threshold, then the write request is added (at block 504) to the write bypass queue 122. The modified tracks of the write request satisfying the condition are buffered (at block 506) in the

buffer 124. Write requests in the write bypass queue 122 are written directly from the buffer 124 to the sequential access storage medium 114 without being cached in the non-volatile storage device 104.

In one embodiment, the condition that determines whether the write request is not cached in the non-volatile storage device 104 and written directly to the sequential access storage medium 114 may be the write request comprising a sequential write request. In an alternative embodiment, modified tracks may be cached in the non-volatile storage device if the block size of the write request is below a threshold, so that sequential and non-sequential write requests having a number of blocks below the threshold are cached in the non-volatile storage device 104.

If (at block 502) the attribute of the write request does not satisfy the condition, e.g., is a non-sequential write request, then the I/O manager 102 adds (at block 508) the received modified tracks to the non-volatile storage device 104. In one embodiment, the tracks may be added to an LSA in the non-volatile storage device 104 or stored in another format in the device 104. The I/O manager 102 creates (at block 510) a cache control block 170 (FIG. 7) for each received modified track indicating a location in the non-volatile storage device 104 (e.g., LSA location) of the modified track. An entry is added (at block 512) to the track index 108 having the track ID of modified track in the non-volatile storage device 104 and index to the created control block 170.

The I/O manager 102 determines (at block 514) a physical location of where the modified track is stored on the sequential access storage medium 114, such as a cylinder on the media. Further, in an additional embodiment, the determined physical location included in the spatial index 112 may also include an angular position on the cylinder of the modified track (also referred to as the sector). The I/O manager 102 adds (at block 516) an entry to the spatial index 112 indicating the track ID 182 of the modified track and the determined physical location 184 of the modified on the sequential access storage medium 114. The I/O manager 102 further adds (at block 518) a destage request to the request queue 116 for each track to write. This destage request may not identify the specific modified track to demote,

which is later determined using an algorithm to reduce the total access time to perform the write.

FIG. 16 illustrates an embodiment of operations performed by the I/O manager 102 to check multiple attributes of the write request to determine whether to queue the write request in the request queue 116 or the write bypass queue 122. Upon initiating (at block 530) the operation to determine whether the write request satisfies the condition, the I/O manager 102 determines (at block 532) whether the received write request is a sequential write request. If (from the no branch of block 532) the write request is non-sequential, then the I/O manager 102 stores (at block 534) the modified tracks for the write request in the non-volatile storage device 104 and adds a destage request at the MRU end of the request queue 116. If (at block 532) the write request is a sequential write request and if (at block 536) the size of the write, such as the number of blocks, exceeds a threshold, i.e., is a large write request, then the I/O manager adds (at block 538) the write request to the write bypass queue 122 and stores the modified tracks in the buffer 124. If (at block 536) the size of the sequential write request does not exceed the threshold, i.e., is a relatively smaller sequential write, then control proceeds to block 534 where that smaller size sequential write is cached in the non-volatile storage device. 104.

FIG. 17 illustrates an embodiment of operations performed by the I/O manager 102 to process the request queue 116 which may be continually repeated while requests are queued in the request queue 116. Upon initiating (at block 550) an operation to process the request queue 116, if (at block 552) the request is a read request, then the I/O manager 102 gathers (at block 554) any of the requested tracks in the non-volatile storage device 104 to return to the read request. If (at block 556) there are requested tracks not in the non-volatile storage device 104, then the I/O manager 102 gathers (at block 558) any of the requested tracks not found in the non-volatile storage device 104 from the sequential access storage medium 114. After gathering all the requested tracks (from block 558 or the no branch of block 556), the I/O manager 102 returns (at block 560) the gathered read requested tracks to the storage controller 4 (FIG. 1) without caching the read requested tracks in the non-volatile storage device 104.

If (at block 562) the request is a destage/write request, then control proceeds (at block 562) to block 600 in FIG. 18 to process the destage/write request. To execute (at block 600) the destage request, the I/O manager 102 compares (at block 602) a current position of the write head 120 with respect to the sequential access storage medium 114 to physical locations (e.g., cylinder and angular position) of the modified tracks indicated in the spatial index 112 and otherwise determined on the sequential access storage medium. The spatial index 112 may include all the necessary information to determine the track in closest temporal proximity to the write head, such as the cylinder and angular position of the track to write, or may include only some of the information, e.g., the cylinder, and the rest of the physical location information needed may be determined from the read/write control unit 118. The I/O manager 102 selects (at block 606), based on the comparison, a modified track that can be written in a minimal time from the current position of the write head 120 and writes (at block 606) the selected modified track to the sequential access storage medium 114. The destaged modified track is invalidated (at block 608).

In an embodiment, where the sequential access storage device comprises a hard disk drive and the sequential access storage medium 114 comprises a magnetic disk, the spatial index indicates a cylinder of the track on magnetic disk. To determine the modified track that can be accessed in the minimal time from the current position of the write head, the I/O manager 102 may analyze the cylinder and angular position of the modified tracks in the spatial index 112 to estimate the times for the write head 120 to seek to the cylinders of the modified tracks and rotate the disk under the write head 120 to reach the angular positions of the modified tracks. The I/O manager may then select a modified track having a minimal of the estimated access times.

In a further embodiment the sequential access storage device 114 may comprise a hard disk drive having multiple disk platters and multiple write heads to write to each platter. The I/O manager 102 may determine the estimated time to seek and rotate to each modified track on each disk platter from the current position of the write heads to select a modified track having the minimal estimated time to access across the disk platters.

FIG. 19 illustrates an embodiment of operations performed by the I/O manager 102 to process write requests in the write bypass queue 122 and process write and read requests in the request queue 116. Upon completing (at block 650) the processing of a read or write request in the request queue 116, the I/O manager 102 determines (at block 652) whether the write bypass request queue is empty. If so, then the I/O manager 102 proceeds (at block 654) to block 550 in FIG. 17 to process a request in the request queue 116. If (at block 652) the write bypass request queue 122 is not empty, then the I/O manager 102 processes (at block 656) a write request in the write bypass queue 122 by writing the buffered modified tracks for the processed write request to the sequential access storage medium 114, bypassing the non-volatile storage device 104. In this way, higher priority is provided to the write requests in the write bypass queue 122 than the request queue 116. After processing a write request from the write bypass queue 122, if (at block 658) a consecutive first predetermined number of write requests in the write bypass queue 122 have been processed, then control proceeds to block 550 in FIG. 17 to process a second predetermined number of requests in the request queue 116 to avoid starvation of the requests in the request queue 116. Otherwise, if (at block 658) the consecutive first predetermined number of write requests from the write bypass queue 122 have not been processed, i.e., the starvation condition has not occurred, then control proceeds back to block 656 to process a further write request in the write bypass queue 122.

In addition, if the I/O manager 102 determines that a destage operation needs to be performed to destage modified tracks in the non-volatile storage device 104 to the sequential access storage medium 114 to free space in the non-volatile storage medium 104, then the destage operation may interrupt the processing of the write requests in the write bypass queue 122 until the destage operation is completed.

Described embodiments provide techniques for allowing the use of a second level cache between a primary or first level cache and a storage to increase the cache space when the fastest access first cache 14 has the most expensive space, cost per byte, and a second cache, less expensive than the first cache but faster than the storage, can be used to increase the amount of cached data in the system. Increasing faster access cached storage space improves access to the cached data when requested data is in the cache and can be returned

from cache instead of having to retrieve from the slower access, less expensive storage. Further, in described embodiments, unmodified non-sequential tracks are added to the second cache based on a temporal ordering in the first cache, and then sorted in the second cache based on spatial physical location in the sequential access storage so that destaged tracks are written in groups of tracks at proximate or consecutive physical locations in the storage to optimize the writing of the tracks to the storage.

Described embodiments further provide a non-volatile storage device 104, such as a flash memory, in the sequential access storage device 100 to allow caching of modified tracks, where read requests to tracks can be returned from the non-volatile storage device 104 before they are destaged to the sequential access medium 114 to improve read performance. Further, write performance may be improved by returning complete to the write in response to the write being stored in the non-volatile storage device 104 before being destaged to the sequential access storage medium 114.

Further benefits are realized by writing certain types of write requests, typically for data that is less frequently accessed, e.g., sequential write data, directly to the sequential access storage medium 114 and bypassing the non-volatile storage device 104 cache to provide more space in the non-volatile storage device 104 cache for write data that is more likely to be frequently accessed, such as non-sequential write tracks. This allows more frequently accessed data to be returned to read requests from the faster access non-volatile storage device 104, such as a flash memory, then having to access the requested more frequently accessed data from the slower sequential access storage medium 114, such as a disk drive.

The described operations may be implemented as a method, apparatus or computer program product using standard programming and/or engineering techniques to produce software, firmware, hardware, or any combination thereof. Accordingly, aspects of the embodiments may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a “circuit,” “module” or “system.” Furthermore, aspects of the embodiments may take the form of a

computer program product embodied in one or more computer readable medium(s) having computer readable program code embodied thereon.

Any combination of one or more computer readable medium(s) may be utilized. The computer readable medium may be a computer readable signal medium or a computer readable storage medium. A computer readable storage medium may be, for example, but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. More specific examples (a non-exhaustive list) of the computer readable storage medium would include the following: an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing. In the context of this document, a computer readable storage medium may be any tangible medium that can contain or store a program for use by or in connection with an instruction execution system, apparatus, or device.

A computer readable signal medium may include a propagated data signal with computer readable program code embodied therein, for example, in baseband or as part of a carrier wave. Such a propagated signal may take any of a variety of forms, including, but not limited to, electro-magnetic, optical, or any suitable combination thereof. A computer readable signal medium may be any computer readable medium that is not a computer readable storage medium and that can communicate, propagate, or transport a program for use by or in connection with an instruction execution system, apparatus, or device.

Program code embodied on a computer readable medium may be transmitted using any appropriate medium, including but not limited to wireless, wireline, optical fiber cable, RF, etc., or any suitable combination of the foregoing.

Computer program code for carrying out operations for aspects of the present invention may be written in any combination of one or more programming languages, including an object oriented programming language such as Java, Smalltalk, C++ or the like and conventional

procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider).

Aspects of the present invention are described above with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer program products according to embodiments of the invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These computer program instructions may be provided to a processor of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

These computer program instructions may also be stored in a computer readable medium that can direct a computer, other programmable data processing apparatus, or other devices to function in a particular manner, such that the instructions stored in the computer readable medium produce an article of manufacture including instructions which implement the function/act specified in the flowchart and/or block diagram block or blocks.

The computer program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other devices to cause a series of operational steps to be performed on the computer, other programmable apparatus or other devices to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

The terms "an embodiment", "embodiment", "embodiments", "the embodiment", "the embodiments", "one or more embodiments", "some embodiments", and "one embodiment" mean "one or more (but not all) embodiments of the present invention(s)" unless expressly specified otherwise.

The terms "including", "comprising", "having" and variations thereof mean "including but not limited to", unless expressly specified otherwise.

The enumerated listing of items does not imply that any or all of the items are mutually exclusive, unless expressly specified otherwise.

The terms "a", "an" and "the" mean "one or more", unless expressly specified otherwise.

Devices that are in communication with each other need not be in continuous communication with each other, unless expressly specified otherwise. In addition, devices that are in communication with each other may communicate directly or indirectly through one or more intermediaries.

A description of an embodiment with several components in communication with each other does not imply that all such components are required. On the contrary a variety of optional components are described to illustrate the wide variety of possible embodiments of the present invention.

Further, although process steps, method steps, algorithms or the like may be described in a sequential order, such processes, methods and algorithms may be configured to work in alternate orders. In other words, any sequence or order of steps that may be described does not necessarily indicate a requirement that the steps be performed in that order. The steps of processes described herein may be performed in any order practical. Further, some steps may be performed simultaneously.

When a single device or article is described herein, it will be readily apparent that more than one device/article (whether or not they cooperate) may be used in place of a single device/article. Similarly, where more than one device or article is described herein (whether or not they cooperate), it will be readily apparent that a single device/article may be used in place of the more than one device or article or a different number of devices/articles may be used instead of the shown number of devices or programs. The functionality and/or the features of a device may be alternatively embodied by one or more other devices which are not explicitly described as having such functionality/features. Thus, other embodiments of the present invention need not include the device itself.

The illustrated operations of FIGs. 7-19 show certain events occurring in a certain order. In alternative embodiments, certain operations may be performed in a different order, modified or removed. Moreover, steps may be added to the above described logic and still conform to the described embodiments. Further, operations described herein may occur sequentially or certain operations may be processed in parallel. Yet further, operations may be performed by a single processing unit or by distributed processing units.

CLAIMS

1. A computer program product for managing data in a sequential access storage device receiving read requests and write requests from a system with respect to tracks stored in a sequential access storage medium, the computer program product comprising a computer readable storage medium having computer readable program code embodied therein that executes to perform operations, the operations comprising:
 - caching received modified tracks in a non-volatile storage device integrated with the sequential access storage device in response to determining to cache the modified tracks;
 - receiving a write request having modified tracks;
 - determining whether an attribute of the received write request satisfies a condition;
 - caching the received modified tracks for the write request in the non-volatile storage device in response to determining that the determined attribute does not satisfy the condition;
 - adding a destage request to a request queue for the received write request having the determined attribute not satisfying the condition; and
 - writing the received modified tracks for the write request having the determined attribute satisfying the condition at a higher priority than modified tracks for write requests having the attribute not satisfying the condition.
2. The computer program product of claim 1, wherein the non-volatile storage device is a faster access device than the sequential access storage medium.
3. The computer program product of claim 1, wherein the sequential access storage device comprises a hard disk drive including a buffer, wherein the non-volatile storage device comprises a flash device separate from a device including the buffer, wherein the sequential access storage medium comprises at least one magnetic disk, wherein the received modified tracks for the write request having the determined attribute satisfying the condition are stored in the buffer until written to the sequential access storage medium without being stored in the non-volatile storage device.
4. The computer program product of claim 1, wherein the operations further comprise:

maintaining a spatial index indicating the modified tracks in the non-volatile storage device in an ordering based on their physical location in the sequential access storage medium; and

in response to processing a destage request in the request queue, comparing a current position of a write head to physical locations of the modified tracks on the sequential access storage medium indicated in the spatial index to select a modified track to destage from the non-volatile storage device to the sequential access medium, and wherein write requests having the determined attribute satisfying the condition are written to the sequential access storage medium without using the spatial index.

5. The computer program product of claim 1, wherein the condition comprises the attribute being a sequential write request and wherein the determined attribute does not satisfy the condition if the write request comprises a non-sequential write request.

6. The computer program product of claim 1, wherein the determined attribute comprises a number of tracks to write as part of the received write request, wherein the determined attribute satisfies the condition if the determined number of tracks to write is greater than a threshold number of tracks and wherein the condition is not satisfied if the determined number of tracks is less than the threshold number of tracks.

7. The computer program product of claim 1, wherein the determined attribute comprises a first attribute and the determined condition comprises a first condition, wherein the operations further comprise:

determining a second attribute of the write request in response to determining that the first attribute satisfies the first condition;

determining whether the second attribute satisfies a second condition;

caching the received modified tracks in the non-volatile storage device in response to determining that the determined second attribute satisfies the second condition; and

adding a destage request to the request queue for the write request having the determined first attribute satisfying the first condition and the second attribute satisfying the second condition, wherein the received modified tracks for the write request having the determined first attribute satisfying the first condition and the determined second attribute

satisfying the second condition are written at a higher priority than modified tracks for write requests having the first attribute not satisfying the first condition and for write requests having the first attribute satisfying the first condition but having the second attribute not satisfying the second condition.

8. The computer program product of claim 7, wherein the determined first attribute indicates whether the write request is a sequential write or a non-sequential write, and wherein the determined first attribute satisfies the first condition if the first attribute is a sequential write request and the first condition is not satisfied if the first attribute is a non-sequential write request, wherein the second attribute comprises a number of tracks to write as part of the received write request, wherein the determined second attribute satisfies the second condition if the determined number of tracks to write is less than a threshold number of tracks and wherein second the condition is not satisfied if the determined number of tracks is greater than the threshold number of tracks.

9. The computer program product of claim 1, wherein the operations further comprise: maintaining a bypass queue queuing write requests for modified tracks having the determined attribute not satisfying the condition;
in response to completing processing of one write request in the request queue, processing one write request in the bypass queue in response to the bypass queue having at least one write request, wherein write requests in the bypass queue are processed at a higher priority over write requests in the request queue.

10. The computer program product of claim 9, further comprising:
switching to processing a first predetermined number of write requests in the request queue after processing a predetermined second number of write requests in the bypass queue.

11. A method for managing data in a sequential access storage device receiving read requests and write requests from a system with respect to tracks stored in a sequential access storage medium, comprising:

caching received modified tracks in a non-volatile storage device integrated with the sequential access storage device in response to determining to cache the modified tracks;
receiving a write request having modified tracks;
determining whether an attribute of the received write request satisfies a condition;
caching the received modified tracks for the write request in the non-volatile storage device in response to determining that the determined attribute does not satisfy the condition;
adding a destage request to a request queue for the received write request having the determined attribute not satisfying the condition; and
writing the received modified tracks for the write request having the determined attribute satisfying the condition at a higher priority than modified tracks for write requests having the attribute not satisfying the condition.

12. The method of claim 11, wherein the sequential access storage device comprises a hard disk drive including a buffer, wherein the non-volatile storage device comprises a flash device separate from a device including the buffer, wherein the sequential access storage medium comprises at least one magnetic disk, wherein the received modified tracks for the write request having the determined attribute satisfying the condition are stored in the buffer until written to the sequential access storage medium without being stored in the non-volatile storage device.

13. The method of claim 11, further comprising:

maintaining a spatial index indicating the modified tracks in the non-volatile storage device in an ordering based on their physical location in the sequential access storage medium; and

in response to processing a destage request in the request queue, comparing a current position of a write head to physical locations of the modified tracks on the sequential access storage medium indicated in the spatial index to select a modified track to destage from the non-volatile storage device to the sequential access medium, and wherein write requests having the determined attribute satisfying the condition are written to the sequential access storage medium without using the spatial index.

14. The method of claim 11, wherein the condition comprises the attribute being a sequential write request and wherein the determined attribute does not satisfy the condition if the write request comprises a non-sequential write request.

15. The method of claim 11, wherein the determined attribute comprises a number of tracks to write as part of the received write request, wherein the determined attribute satisfies the condition if the determined number of tracks to write is greater than a threshold number of tracks and wherein the condition is not satisfied if the determined number of tracks is less than the threshold number of tracks.

16. The method of claim 11, wherein the determined attribute comprises a first attribute and the determined condition comprises a first condition, further comprising:

determining a second attribute of the write request in response to determining that the first attribute satisfies the first condition;

determining whether the second attribute satisfies a second condition;

caching the received modified tracks in the non-volatile storage device in response to determining that the determined second attribute satisfies the second condition; and

adding a destage request to the request queue for the write request having the determined first attribute satisfying the first condition and the second attribute satisfying the second condition, wherein the received modified tracks for the write request having the determined first attribute satisfying the first condition and the determined second attribute satisfying the second condition are written at a higher priority than modified tracks for write requests having the first attribute not satisfying the first condition and for write requests having the first attribute satisfying the first condition but having the second attribute not satisfying the second condition.

17. The method of claim 16, wherein the determined first attribute indicates whether the write request is a sequential write or a non-sequential write, and wherein the determined first attribute satisfies the first condition if the first attribute is a sequential write request and the first condition is not satisfied if the first attribute is a non-sequential write request, wherein the second attribute comprises a number of tracks to write as part of the received write request, wherein the determined second attribute satisfies the second condition if the

determined number of tracks to write is less than a threshold number of tracks and wherein second the condition is not satisfied if the determined number of tracks is greater than the threshold number of tracks.

18. The computer program product of claim 11, further comprising:

maintaining a bypass queue queuing write requests for modified tracks having the determined attribute not satisfying the condition;

in response to completing processing of one write request in the request queue, processing one write request in the bypass queue in response to the bypass queue having at least one write request, wherein write requests in the bypass queue are processed at a higher priority over write requests in the request queue.

19. A storage device receiving read requests and write requests for tracks from a system, comprising:

a sequential access storage medium storing tracks;

a non-volatile storage device; and

a computer readable storage medium having computer readable program code embodied therein that executes to perform operations, the operations comprising:

caching received modified tracks in the non-volatile storage device in response to determining to cache the modified tracks;

receiving a write request having modified tracks;

determining whether an attribute of the received write request satisfies a condition;

caching the received modified tracks for the write request in the non-volatile storage device in response to determining that the determined attribute does not satisfy the condition;

adding a destage request to a request queue for the received write request having the determined attribute not satisfying the condition; and

writing the received modified tracks for the write request having the determined attribute satisfying the condition at a higher priority than modified tracks for write requests having the attribute not satisfying the condition.

20. The storage device of claim 19, wherein the sequential access storage device comprises a hard disk drive including a buffer, wherein the non-volatile storage device

comprises a flash device separate from a device including the buffer, wherein the sequential access storage medium comprises at least one magnetic disk, wherein the received modified tracks for the write request having the determined attribute satisfying the condition are stored in the buffer until written to the sequential access storage medium without being stored in the non-volatile storage device.

21. The storage device of claim 19, wherein the operations further comprise:

maintaining a spatial index indicating the modified tracks in the non-volatile storage device in an ordering based on their physical location in the sequential access storage medium; and

in response to processing a destage request in the request queue, comparing a current position of a write head to physical locations of the modified tracks on the sequential access storage medium indicated in the spatial index to select a modified track to destage from the non-volatile storage device to the sequential access medium, and wherein write requests having the determined attribute satisfying the condition are written to the sequential access storage medium without using the spatial index.

22. The storage device of claim 19, wherein the condition comprises the attribute being a sequential write request and wherein the determined attribute does not satisfy the condition if the write request comprises a non-sequential write request.

23. The storage device of claim 19, wherein the determined attribute comprises a number of tracks to write as part of the received write request, wherein the determined attribute satisfies the condition if the determined number of tracks to write is greater than a threshold number of tracks and wherein the condition is not satisfied if the determined number of tracks is less than the threshold number of tracks.

24. The storage device of claim 19, wherein the determined attribute comprises a first attribute and the determined condition comprises a first condition, wherein the operations further comprise:

determining a second attribute of the write request in response to determining that the first attribute satisfies the first condition;

determining whether the second attribute satisfies a second condition;
caching the received modified tracks in the non-volatile storage device in response to determining that the determined second attribute satisfies the second condition; and
adding a destage request to the request queue for the write request having the determined first attribute satisfying the first condition and the second attribute satisfying the second condition, wherein the received modified tracks for the write request having the determined first attribute satisfying the first condition and the determined second attribute satisfying the second condition are written at a higher priority than modified tracks for write requests having the first attribute not satisfying the first condition and for write requests having the first attribute satisfying the first condition but having the second attribute not satisfying the second condition.

25. The storage device of claim 24, wherein the determined first attribute indicates whether the write request is a sequential write or a non-sequential write, and wherein the determined first attribute satisfies the first condition if the first attribute is a sequential write request and the first condition is not satisfied if the first attribute is a non-sequential write request, wherein the second attribute comprises a number of tracks to write as part of the received write request, wherein the determined second attribute satisfies the second condition if the determined number of tracks to write is less than a threshold number of tracks and wherein second the condition is not satisfied if the determined number of tracks is greater than the threshold number of tracks.

26. The storage device of claim 19, wherein the operations further comprise:
maintaining a bypass queue queuing write requests for modified tracks having the determined attribute not satisfying the condition;
in response to completing processing of one write request in the request queue, processing one write request in the bypass queue in response to the bypass queue having at least one write request, wherein write requests in the bypass queue are processed at a higher priority over write requests in the request queue.

1/14

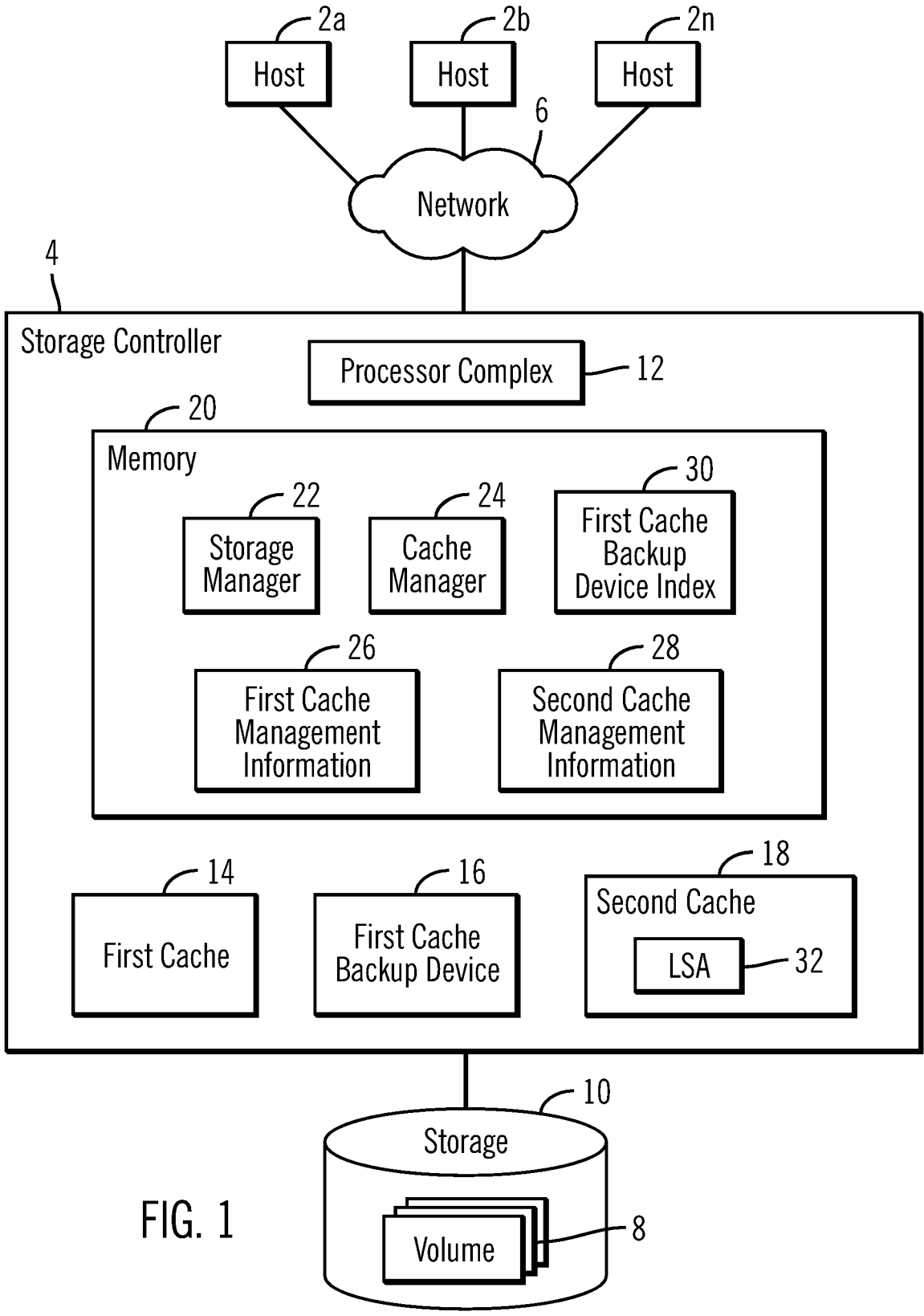


FIG. 1

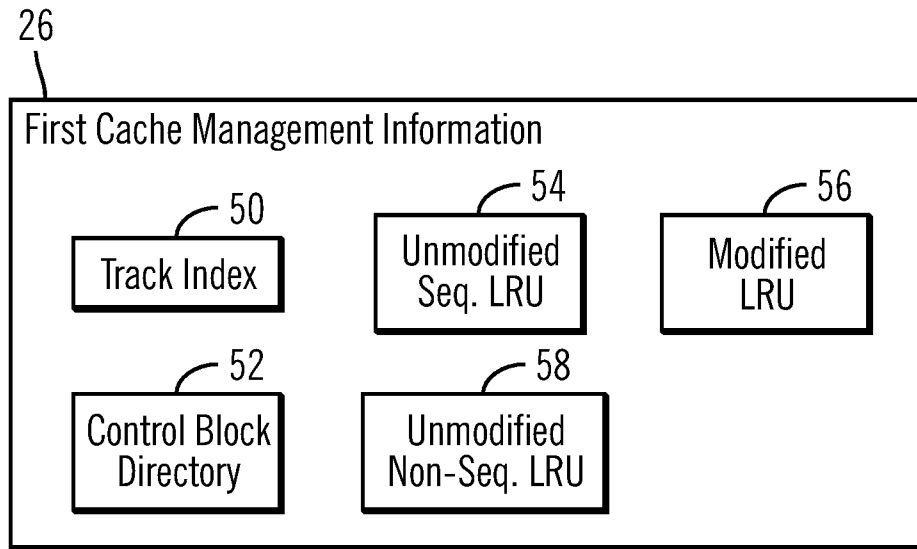


FIG. 2

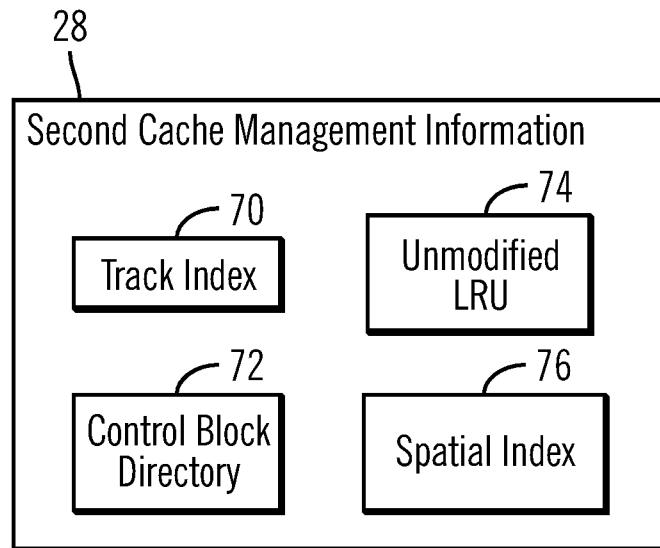


FIG. 3

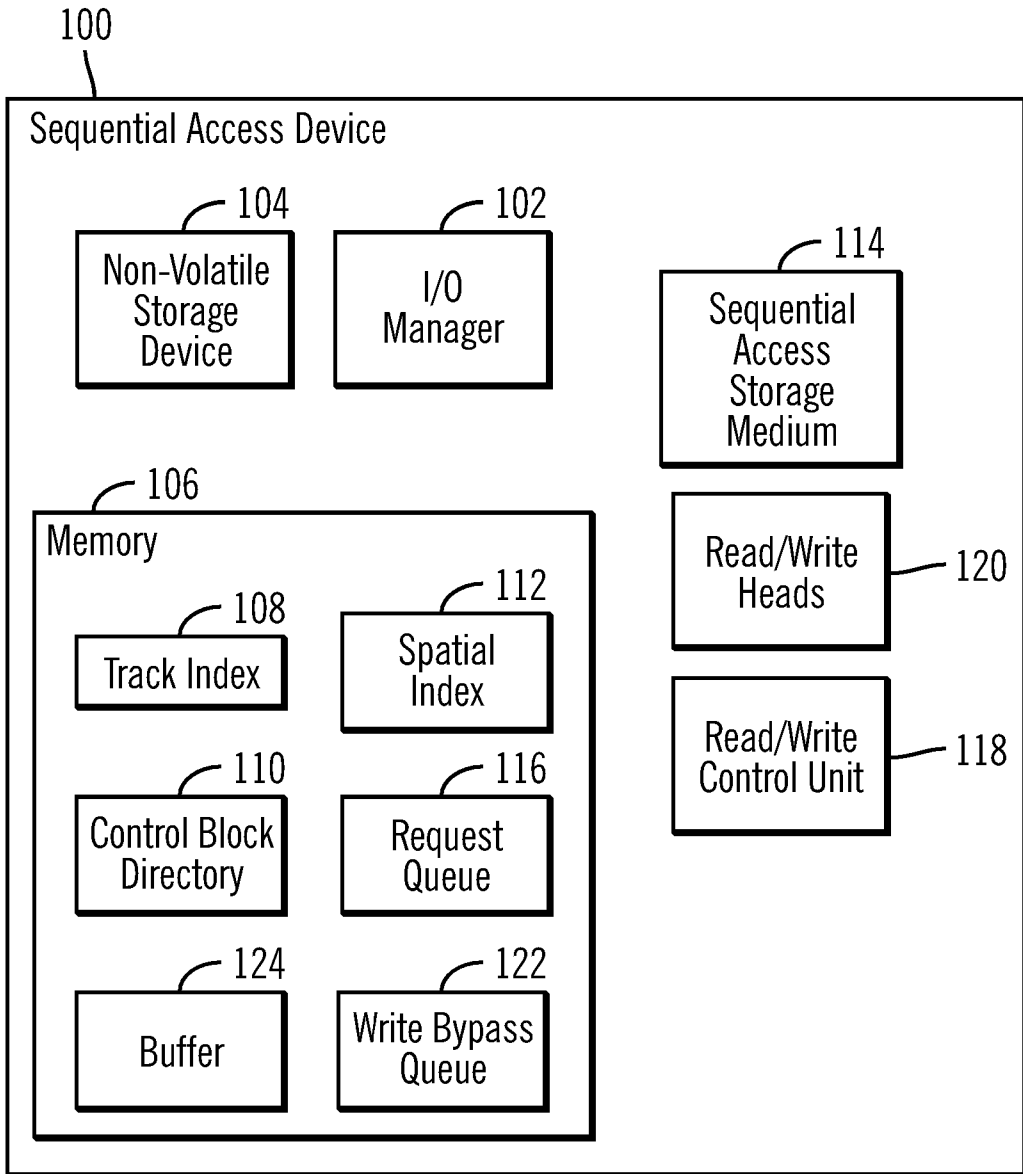
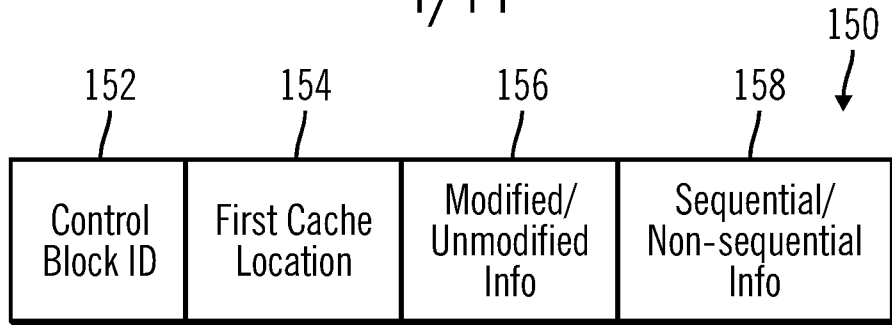


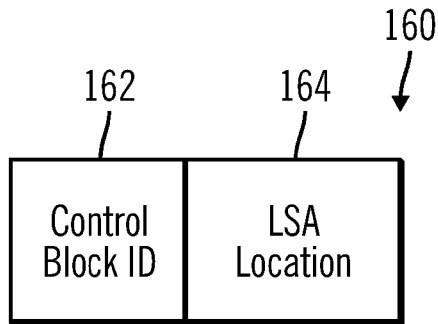
FIG. 4

4/14



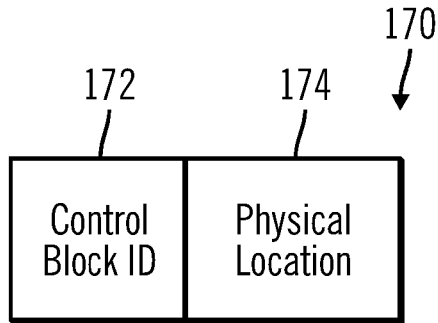
First Cache Control Block

FIG. 5



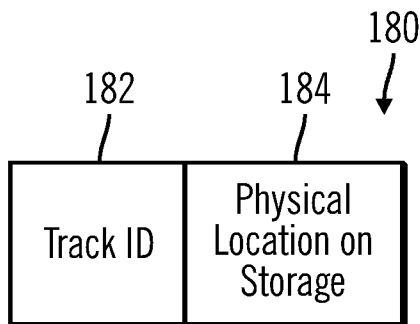
Second Cache Control Block

FIG. 6



Non-Volatile Storage Control Block

FIG. 7



Spatial Index Entry

FIG. 8

5/14

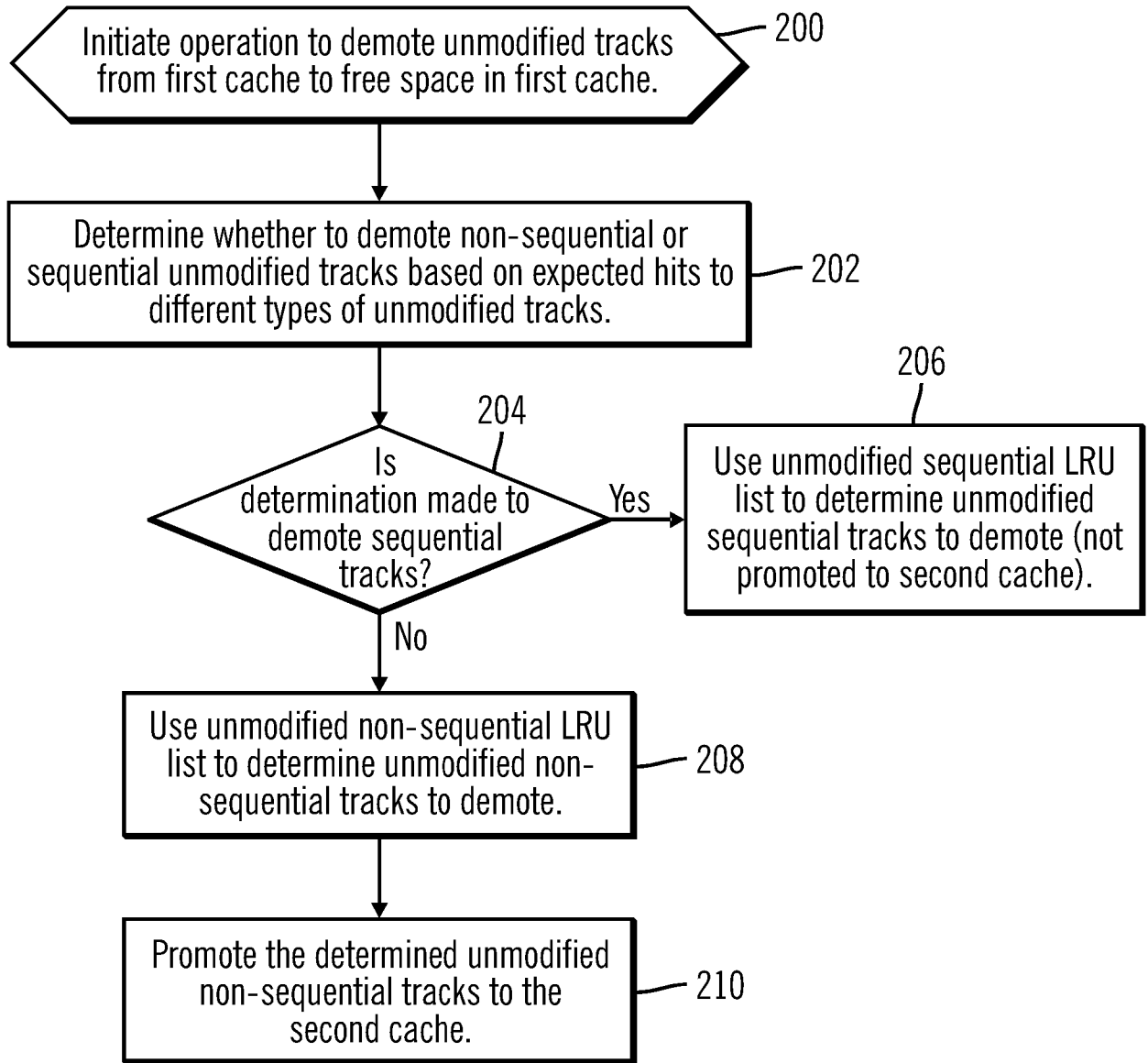


FIG. 9

6/14

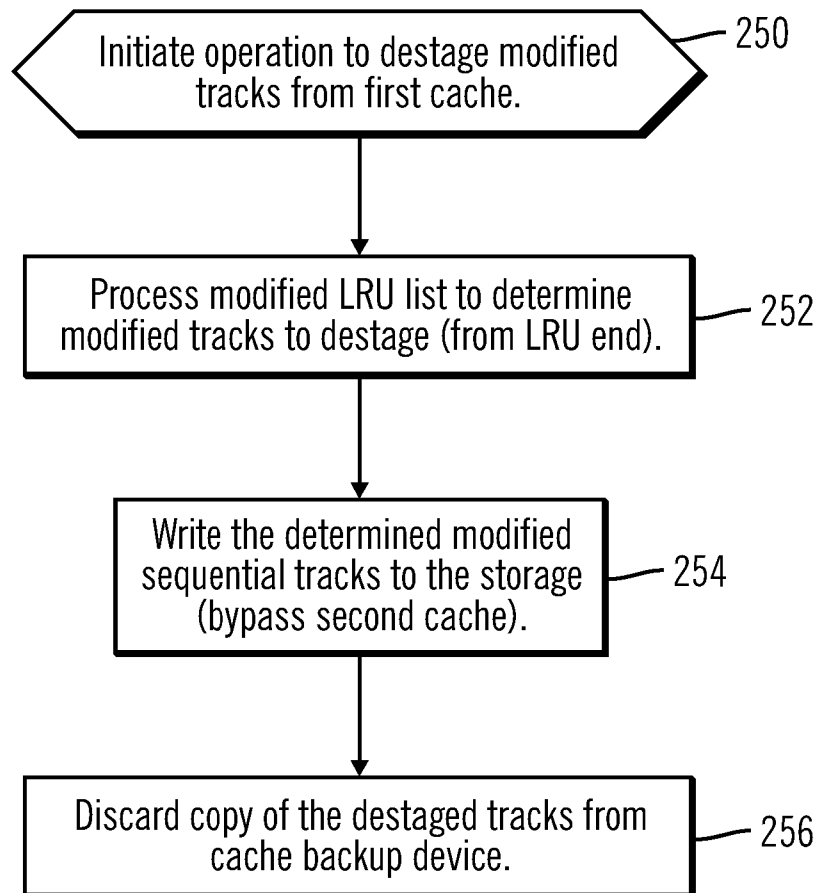


FIG. 10

7/14

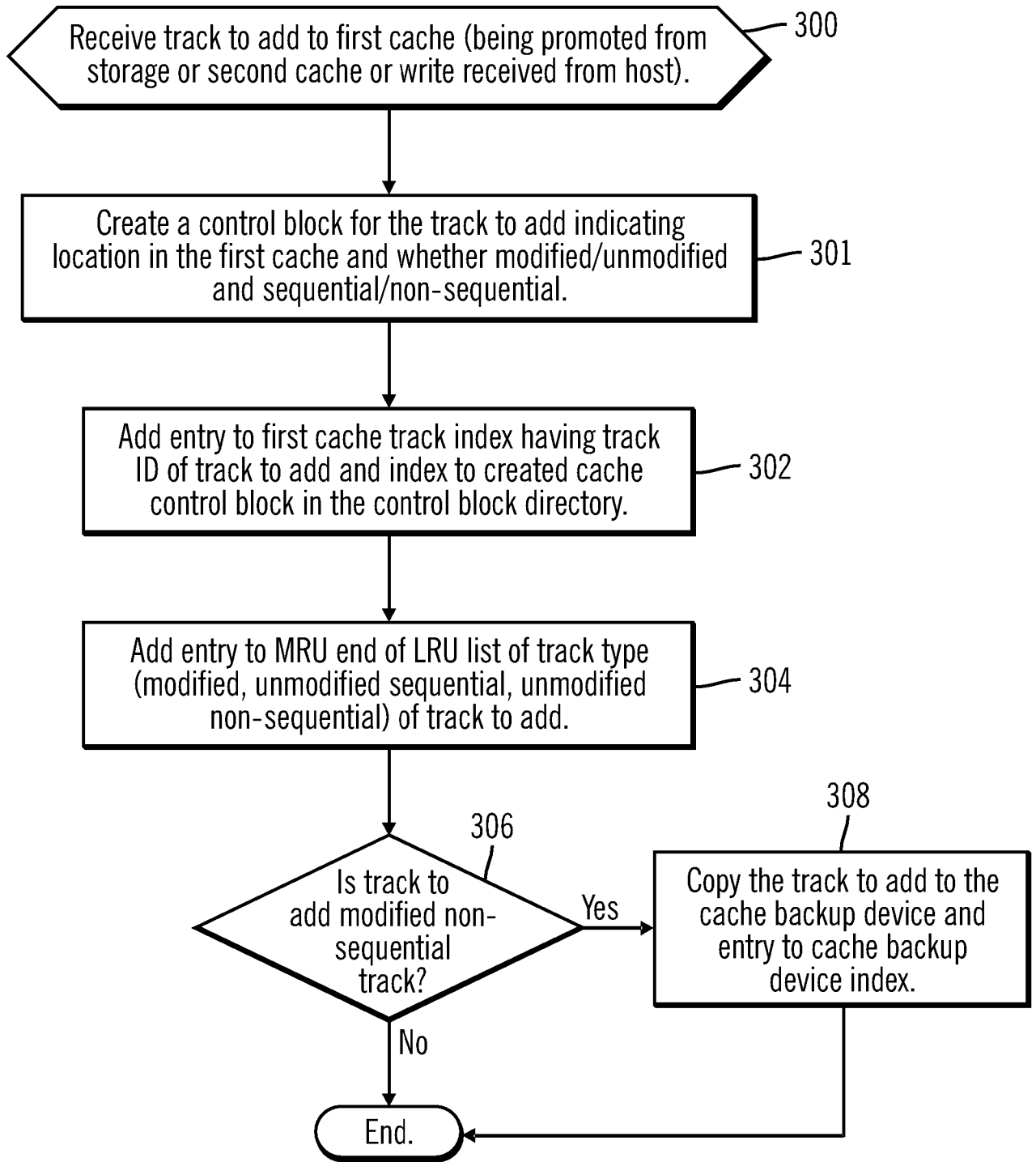


FIG. 11

8/14

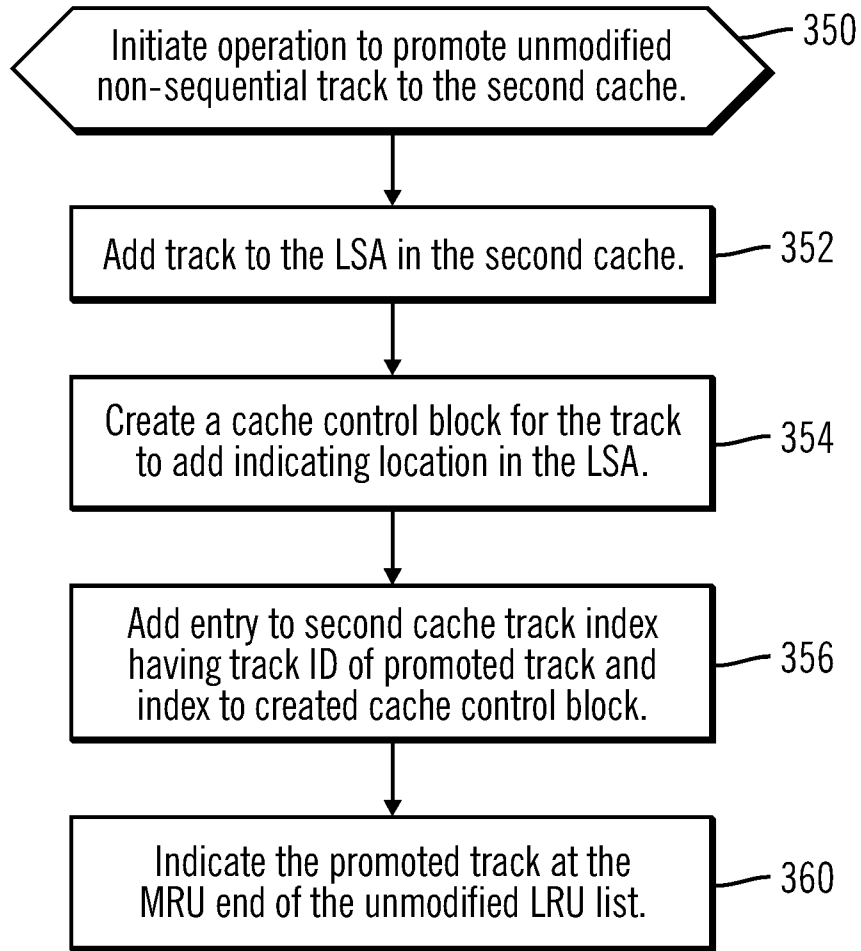


FIG. 12

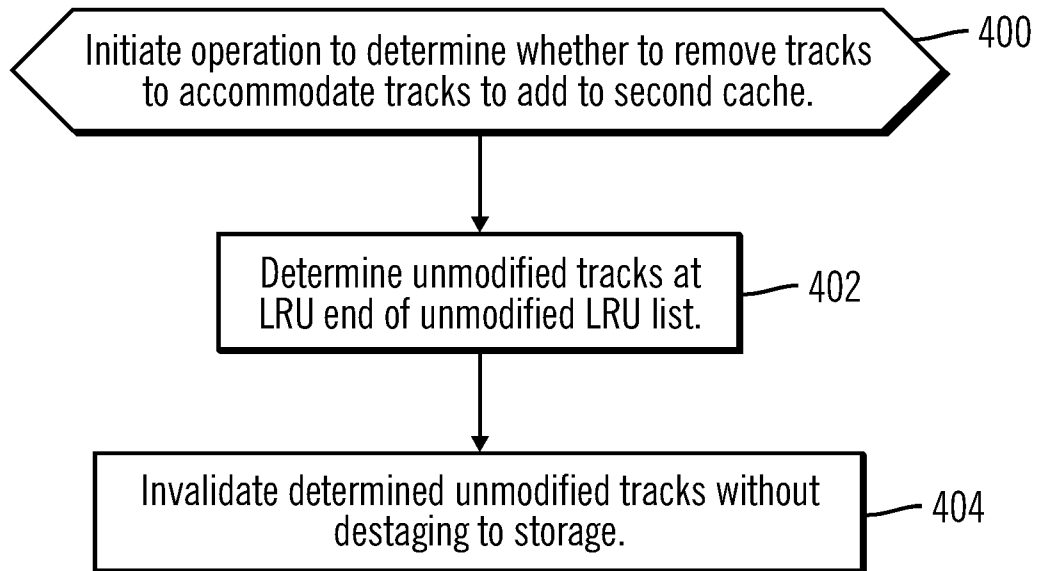


FIG. 13

9/14

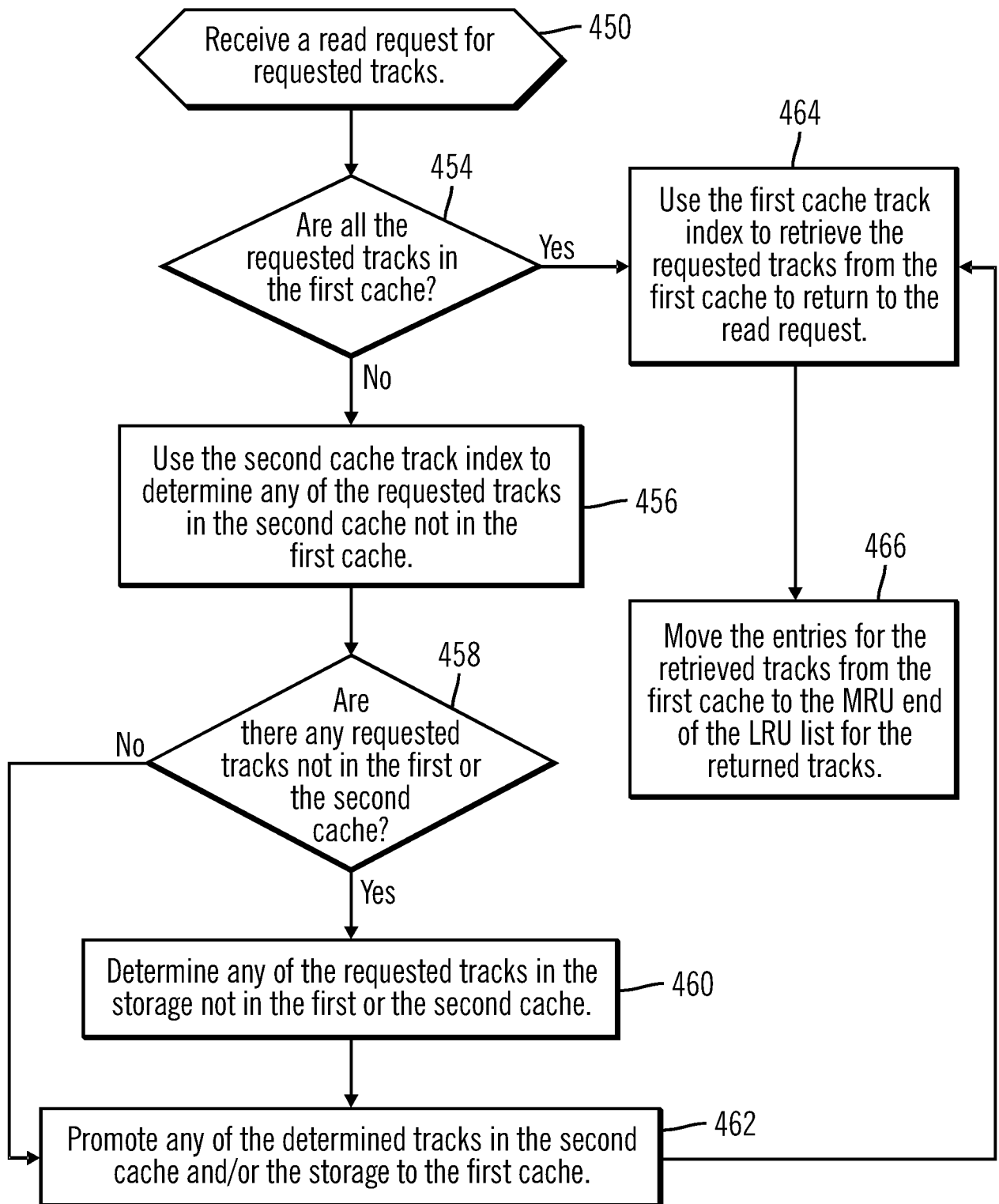


FIG. 14

10/14

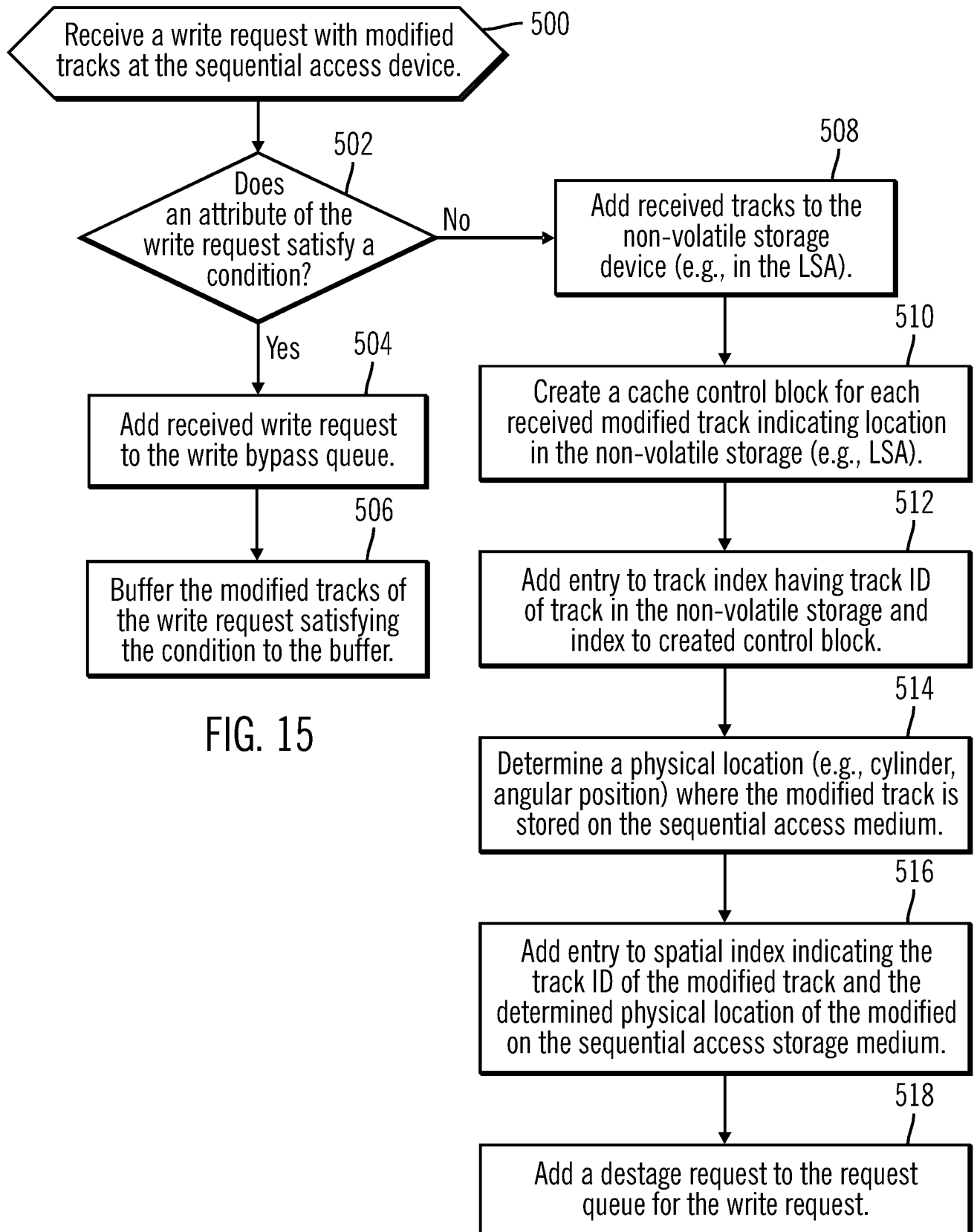


FIG. 15

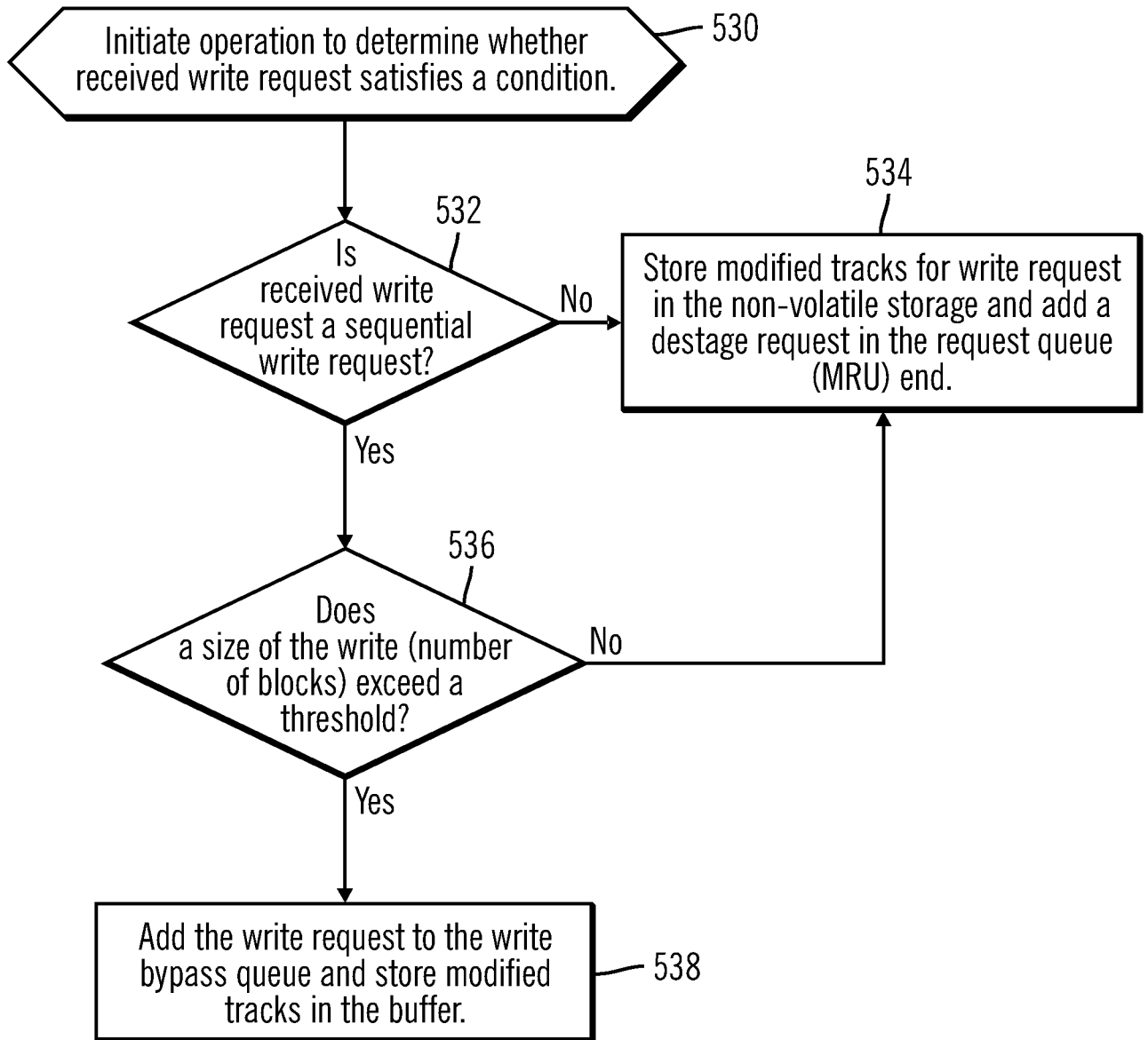


FIG. 16

12/14

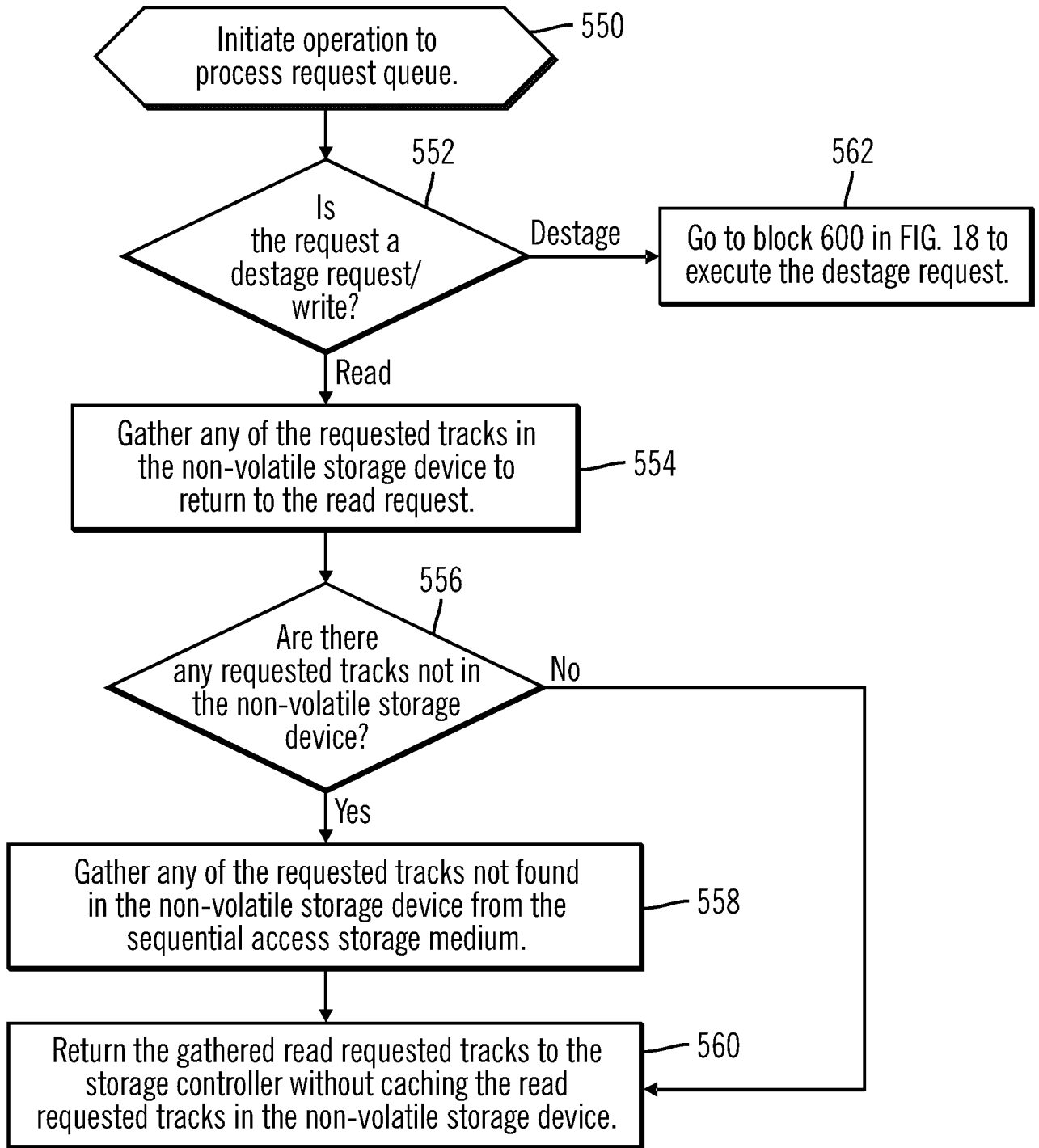


FIG. 17

13/14

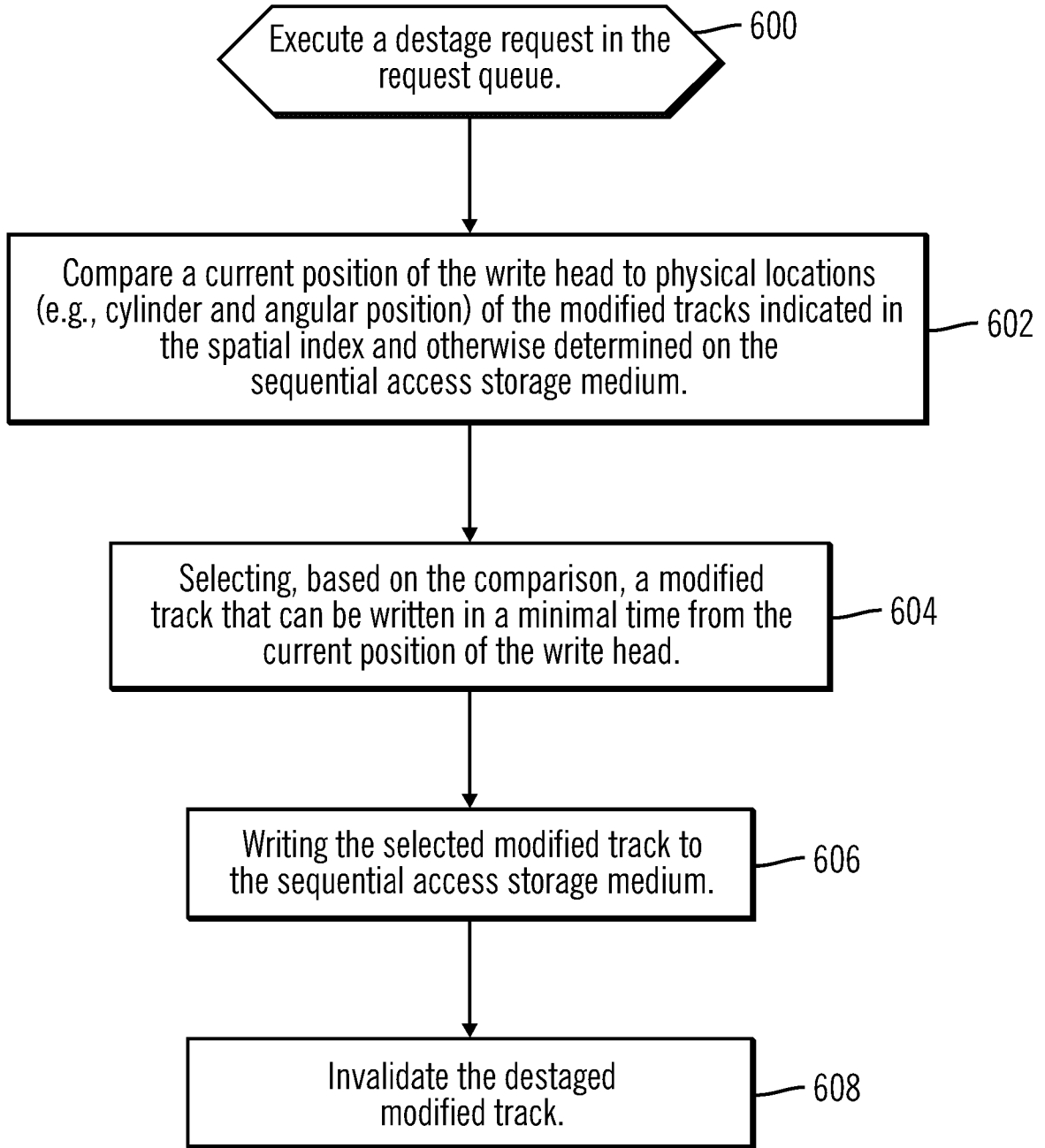


FIG. 18

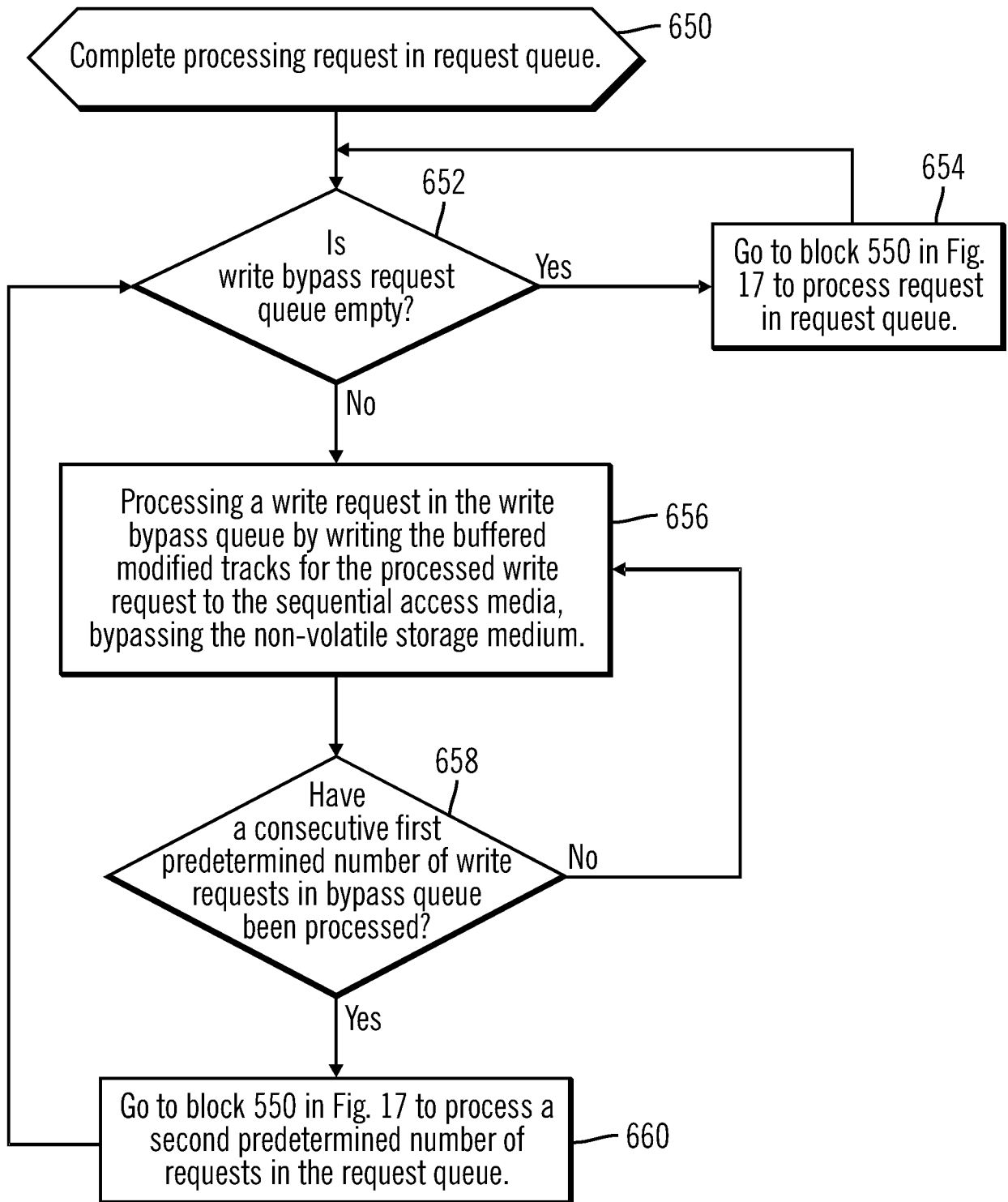


FIG. 19

INTERNATIONAL SEARCH REPORT

International application No.

PCT/IB2012/052564

A. CLASSIFICATION OF SUBJECT MATTER		
Int.Cl. G06F3/06(2006.01) i, G06F12/00(2006.01) i		
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED		
Minimum documentation searched (classification system followed by classification symbols)		
Int.Cl. G06F3/06, G06F12/00		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Published examined utility model applications of Japan 1922-1996 Published unexamined utility model applications of Japan 1971-2012 Registered utility model specifications of Japan 1996-2012 Published registered utility model applications of Japan 1994-2012		
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2008/0168220 A1 (INTERNATIONAL BUSINESS MACHINES CORPORATION) 2008.07.10,	1-5, 11-14, 19-22
A	[0035]-[0076], FIG.1-FIG.12 & CN 101221537 A	6-10, 15-18, 23-26
Y	JP 2007-141225 A (INTERNATIONAL BUSINESS MACHINES CORPORATION) 2007.06.07,	1-5, 11-14, 19-22
A	[0003]-[0016], FIG.1-FIG.23 & US 2007/0118695 A1 & CN 1967507 A	6-10, 15-18, 23-26
A	JP 06-505584 A (ZITEL CORPORATION) 1994.06.23, line 5-13, down-right-side column, page9, line 1-5, down-left-side column, page14, line15, up-left-side column to line21, down-left-side column, page15 & US 5325509 A & EP 574531 A & WO 1992/015933 A1 & AU 1578092 A & CA 2105085 A	1-26
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: “A” document defining the general state of the art which is not considered to be of particular relevance “E” earlier application or patent but published on or after the international filing date “L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) “O” document referring to an oral disclosure, use, exhibition or other means “P” document published prior to the international filing date but later than the priority date claimed “T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention “X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone “Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art “&” document member of the same patent family		
Date of the actual completion of the international search		Date of mailing of the international search report
07.09.2012		18.09.2012
Name and mailing address of the ISA/JP		Authorized officer
Japan Patent Office		Takatoshi KIMURA
3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan		5T 9857
		Telephone No. +81-3-3581-1101 Ext. 3568